UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/901,368	07/09/2001	Nithyalakshmi Sampathkumar	MS180587.1	6483

27195 7590 04/30/2009

TUROCY & WATSON, LLP
127 Public Square
57th Floor, Key Tower
CLEVELAND, OH 44114

EXAMINER	
HILLERY, NATHAN	

ART UNIT	PAPER NUMBER
2176	

NOTIFICATION DATE	DELIVERY MODE
04/30/2009	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

docket1@thepatentattorneys.com
hholmes@thepatentattorneys.com
lpasterchek@thepatentattorneys.com

UNITED STATES PATENT AND TRADEMARK OFFICE

Commissioner for Patents
United States Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450
www.uspto.gov

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

Application Number: 09/901,368
Filing Date: July 09, 2001
Appellant(s): SAMPATHKUMAR ET AL.

Himanshu S. Amin
For Appellant

EXAMINER'S ANSWER

This is in response to the appeal brief filed 1/30/09 appealing from the Office action mailed 5/1/08.

(1) Real Party in Interest

A statement identifying by name the real party in interest is contained in the brief.

(2) Related Appeals and Interferences

The examiner is not aware of any related appeals, interferences, or judicial proceedings which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

(3) Status of Claims

The statement of the status of claims contained in the brief is correct.

(4) Status of Amendments After Final

The appellant's statement of the status of amendments after final rejection contained in the brief is correct.

(5) Summary of Claimed Subject Matter

The summary of claimed subject matter contained in the brief is correct.

(6) Grounds of Rejection to be Reviewed on Appeal

The appellant's statement of the grounds of rejection to be reviewed on appeal is correct.

(7) Claims Appendix

The copy of the appealed claims contained in the Appendix to the brief is correct.

(8) Evidence Relied Upon

6772413	Kuznetsov	8-2004
20030126136	Omoigui	

ADO.NET; Matjaz Klandar in Dejan Sarka; Kopenhagen; 6-8 March 2001 <http://sql. reproms.si/data%5Cpodatki%5CMatjazk%5Cado, net. ppt>, (English Translation)

(9) Grounds of Rejection

The following ground(s) of rejection are applicable to the appealed claims:

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically teach that or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

Claim 1, 3, 4, 5 and 19 are rejected under 35 U.S.C. 103(a) as being unpatentable over Kuznetsov (US 6772413 B2).

Regarding independent claim 1, Kuznetsov teaches that the generalized system can be applied to the growing problem of integrating disparate or incompatible computer systems, file formats, network protocols, or other machine data. This applies whether the data is recorded in a storage device, preserved in temporary memory, or transmitted over a network. This approach allows many more formats and protocols to be accommodated flexibly while preserving the performance and simplification advantages (Column 9, lines 25 – 33), which meet the limitation of **a memory configured to receive and store one or more input XML items**.

Kuznetsov teaches that to transform an input XML vocabulary to another (output) XML vocabulary, the XSLT translator processor must parse the transform, parse the source data, walk the two parse trees to apply the transform, and finally output the data

into a stream (Column 14, lines 51 – 59), which meets the limitation of **a transformer that transforms one or more input XML items in a first format to one or more transformed XML items in one or more second XML formats**.

Kuznetsov teaches that any number of translators can be implemented simultaneously, such that an entire set (or selected subset) of packets can be translated during runtime (Column 13, line 66 – column 14, line 1), which meets the limitation of **an output manager that facilitates at least one of selectively pulling and pushing a subset of the one or more input XML items**.

Kuznetsov does not explicitly say that **the subset of the one or more XML items is less than the whole one or more input XML items**.

However Kuznetsov teaches an entire set or selected subset (Column 13, line 66 – Column 14, line 1) as two different entities. Thus, it would have been to a person of ordinary skill in the art to try translating a selected subset that is less than the entire set in an attempt to provide an improved translation, as a person with ordinary skill has good reason to pursue the known options within his or her technical grasp. In turn, because the subset as claimed has the properties predicted by the prior art, it would have been obvious to translate a selected subset that is less than the entire set.

Regarding dependent claim 3, Kuznetsov teaches that a data translator compiler is adapted for using the XSL stylesheet as its input. The data translator compiler then generates executable machine code that operates as a run-time translator between the source XML and the target XML (Column 14, line 60 – Column

15, line 2), which meets the limitation of **a compiler that compiles one or more style sheets and produce one or more actions that can be employed by the transformer in processing associated with transforming the one or more input XML items**.

Regarding dependent claim 4, Kuznetsov teaches that an implementation according the present invention may also incorporate predefined functions, or references to external functions that can be called at runtime, according to the needs of the translator, as generated by the translator compiler engine (Column 14, lines 23 – 27), which meets the limitation of **the compiler resolves one or more external references in the one or more style sheets**.

Regarding dependent claim 5, Kuznetsov teaches that whether the data is recorded in a storage device, preserved in temporary memory, or transmitted over a network, the approach allows many more formats and protocols to be accommodated flexibly while preserving the performance and simplification advantages (Column 9, lines 28 – 33), which meets the limitation of **the input XML items are input from one or more data stores**.

Regarding independent claim 19, Kuznetsov teaches that the generalized system can be applied to the growing problem of integrating disparate or incompatible computer systems, file formats, network protocols, or other machine data. This applies whether the data is recorded in a storage device, preserved in temporary memory, or

transmitted over a network. This approach allows many more formats and protocols to be accommodated flexibly while preserving the performance and simplification advantages (Column 9, lines 25 – 33), which meet the limitation of **a memory configured to receive and store one or more input XML items**.

Kuznetsov teaches that to transform an input XML vocabulary to another (output) XML vocabulary, the XSLT translator processor must parse the transform, parse the source data, walk the two parse trees to apply the transform, and finally output the data into a stream (Column 14, lines 51 – 59), which meets the limitation of **a transforming component that transforms an input XML item from a first format to a transformed XML item in one or more second XML formats**.

Kuznetsov teaches that any number of translators can be implemented simultaneously, such that an entire set (or selected subset) of packets can be translated during runtime (Column 13, line 66 – column 14, line 1), which meets the limitation of **an output managing component that facilitates at least one of selectively pulling and pushing a subset of the input XML item, the subset of the one or more XML items is less than the whole input XML item**.

Kuznetsov teaches that a data translator compiler is adapted for using the XSL stylesheet as its input. The data translator compiler then generates executable machine code that operates as a run-time translator between the source XML and the target XML (Column 14, line 60 – Column 15, line 2), which meets the limitation of **a compiling component that compiles a style sheet and that produces one or more actions**

that can be employed by the transforming component in processing associated with transforming the input XML item.

Kuznetsov teaches that whether the data is recorded in a storage device, preserved in temporary memory, or transmitted over a network, the approach allows many more formats and protocols to be accommodated flexibly while preserving the performance and simplification advantages (Column 9, lines 28 – 33) and that the optimization options comprise first optimization pass, which generates intermediate format, and second optimization pass (Column 16, lines 49 – 54), which meets the limitation of **an input abstracting component that presents input XML items stored in one or more different representations to the transforming component in a common representation**.

Kuznetsov teaches that as currently specified by the Worldwide Web Consortium, there are three major components in an XSL processor: XSLT, the transformation engine; Xpath, the node selection and query module; and Formatting Objects, the formatting and end-user presentation layer specification. XML-to-XML data translation is primarily concerned with the first two modules (Column 14, lines 33 – 39), which meets the limitation of **a node selection abstracting component that dynamically constructs a subset of input XML items from a set of input XML items, the subset of input XML items are responsive to a query**.

Claim 2 is rejected under 35 U.S.C. 103(a) as being unpatentable over Kuznetsov (US 6772413 B2) as applied to claim 1 above and further in view of Omoigui (US 20030126136 A1).

Regarding dependent claim 2, Kuznetsov does not explicitly teach that **the transformer comprises an action frame stack that holds one or more actions, an event state machine that tracks state associated with transforming the one or more input XML items and an event processor that receives events generated in processing the one or more actions stored in the action frame stack**.

However, Omoigui teaches that the system provides support for authentication, authorization, auditing, data privacy, data integrity, availability, and non-repudiation by employing standards such as WS-Security. WS-Security provides a platform for security with XML Web Service applications using standards in the XML Web Service protocol stack. This includes encrypting method calls from clients, support for digital signatures, authenticating the calling user before granting access to an Agency's Semantic Network and XML Web Service methods, etc. (paragraph block 0367), which meets the limitation of **the transformer comprises an action frame stack that holds one or more actions, an event state machine that tracks state associated with transforming the one or more input XML items and an event processor that receives events generated in processing the one or more actions stored in the action frame stack**.

It would have been obvious to one of ordinary skill in the art at the time of the invention to combine the teachings of Kuznetsov with the invention of Omoigui because such a combination would provide the readers of Kuznetsov with *an integrated and seamless implementation framework and resulting medium for knowledge retrieval, management, delivery and presentation* (paragraph block 0071).

Claims 6 – 18 and 38 are rejected under 35 U.S.C. 103(a) as being unpatentable over Kuznetsov (US 6772413 B2) as applied to claim 1 above and further in view of ADO.NET (English translation).

Regarding dependent claim 6, Kuznetsov does not explicitly teach that **an input abstracter that exposes data stored in the one or more data stores in a common representation**.

ADO.NET teach that an XpathNavigator is created to abstract data from the xml data set via an XpathNodeIterator by employing a loop (p 19), which meets the limitation of **an input abstracter that exposes data stored in the one or more data stores in a common representation**.

It would have been obvious to one of ordinary skill in the art at the time of the invention to combine the teachings of Kuznetsov with those of ADO.NET because such a combination would provide the users of Kuznetsov with the benefit of explicit implementation of XPath via source code.

Regarding dependent claim 7, Kuznetsov does not explicitly teach that **the input abstractor abstracts a reference to a node within an Xpath document**.

ADO.NET teach that an XpathNavigator is created to abstract data from the xml data set via an XpathNodeIterator (p 19), which meets the limitation of **the input abstractor abstracts a reference to a node within an Xpath document**.

It would have been obvious to one of ordinary skill in the art at the time of the invention to combine the teachings of Kuznetsov with those of ADO.NET because such

a combination would provide the users of Kuznetsov with the benefit of explicit implementation of XPath via source code.

Regarding dependent claims 8, Kuznetsov does not explicitly teach that **the input abstractor exposes the data stored in the one or more data stores as a data model and infoset**.

ADO.NET teach that an XpathNavigator is created to abstract data from the xml data set (p 19), which meets the limitation of **the input abstractor exposes the data stored in the one or more data stores as a data model and infoset**.

It would have been obvious to one of ordinary skill in the art at the time of the invention to combine the teachings of Kuznetsov with those of ADO.NET because such a combination would provide the users of Kuznetsov with the benefit of explicit implementation of XPath via source code.

Regarding dependent claim 9, Kuznetsov does not explicitly teach that **the input abstractor provides a cursor model over data stored in a data store to facilitate presenting a stream of nodes to the transformer**.

ADO.NET teach that an XpathNavigator is created to abstract data from the xml data set and sends the data to an XSLT (p 19), which meets the limitation of **the input abstractor provides a cursor model over data stored in a data store to facilitate presenting a stream of nodes to the transformer**.

It would have been obvious to one of ordinary skill in the art at the time of the invention to combine the teachings of Kuznetsov with those of ADO.NET because such a combination would provide the users of Kuznetsov with the benefit of explicit implementation of XPath via source code.

Regarding dependent claim 10, Kuznetsov does not explicitly teach that **the input abstractor provides a virtual node that can be employed to traverse the stream of nodes**.

ADO.NET teach that an XpathNavigator is created to abstract data from the xml data set (p 19), which meets the limitation of **the input abstractor provides a virtual node that can be employed to traverse the stream of nodes**.

It would have been obvious to one of ordinary skill in the art at the time of the invention to combine the teachings of Kuznetsov with those of ADO.NET because such a combination would provide the users of Kuznetsov with the benefit of explicit implementation of XPath via source code.

Regarding dependent claim 11, Kuznetsov does not explicitly teach that **the input abstractor is an XpathNavigator**.

ADO.NET teach that an XpathNavigator is created to abstract data from the xml data set (p 19), which meets the limitation of **the input abstractor is an XpathNavigator**.

It would have been obvious to one of ordinary skill in the art at the time of the invention to combine the teachings of Kuznetsov with those of ADO.NET because such a combination would provide the users of Kuznetsov with the benefit of explicit implementation of XPath via source code.

Regarding dependent claim 12, Kuznetsov does not explicitly teach that **a node selection abstractor that dynamically constructs a subset of input XML items from a set of input XML items, the subset of input XML items are responsive to a query**.

ADO.NET teach that SQL is used to query xml items and store them to an XML data set and that each node in the xml dataset is visited by employing an XpathNodeIterator (pp 18 – 19), which meets the limitation of **a node selection abstractor that dynamically constructs a subset of input XML items from a set of input XML items, the subset of input XML items are responsive to a query**.

It would have been obvious to one of ordinary skill in the art at the time of the invention to combine the teachings of Kuznetsov with those of ADO.NET because such a combination would provide the users of Kuznetsov with the benefit of explicit implementation of XPath via source code.

Regarding dependent claim 13, Kuznetsov does not explicitly teach that **the node selection abstractor facilitates navigating the subset of input XML items**.

ADO.NET teach that each node in the xml dataset is visited by employing an XpathNodeIterator (pp 18 – 19), which meets the limitation of **the node selection abstractor facilitates navigating the subset of input XML items**.

It would have been obvious to one of ordinary skill in the art at the time of the invention to combine the teachings of Kuznetsov with those of ADO.NET because such a combination would provide the users of Kuznetsov with the benefit of explicit implementation of XPath via source code.

Regarding dependent claim 14, Kuznetsov does not explicitly teach that **the node selection abstractor is an XpathNodeIterator**.

ADO.NET teach that each node in the xml dataset is visited by employing an XpathNodeIterator (pp 18 – 19), which meets the limitation of **the node selection abstractor is an XpathNodeIterator**.

It would have been obvious to one of ordinary skill in the art at the time of the invention to combine the teachings of Kuznetsov with those of ADO.NET because such a combination would provide the users of Kuznetsov with the benefit of explicit implementation of XPath via source code.

Regarding dependent claim 15, Kuznetsov does not explicitly teach that **an optimized data store that stores one or more XML items in a manner that facilitates minimizing processing associated with constructing the subset of input XML items via a query**.

ADO.NET teach that SQL is used to query xml items and store them to an XML data set (pp 18 – 19), which meets the limitation of **an optimized data store that stores one or more XML items in a manner that facilitates minimizing processing associated with constructing the subset of input XML items via a query**.

It would have been obvious to one of ordinary skill in the art at the time of the invention to combine the teachings of Kuznetsov with those of ADO.NET because such a combination would provide the users of Kuznetsov with the benefit of explicit implementation of XPath via source code.

Regarding dependent claim 16, Kuznetsov does not explicitly teach that **the optimized data store stores data in a data representation format that facilitates optimizing an Xpath query**.

ADO.NET teach that Xpath document is created and used to store and manipulate the xml data set (pp 18 – 19), which meets the limitation of **the optimized data store stores data in a data representation format that facilitates optimizing an Xpath query**.

It would have been obvious to one of ordinary skill in the art at the time of the invention to combine the teachings of Kuznetsov with those of ADO.NET because such a combination would provide the users of Kuznetsov with the benefit of explicit implementation of XPath via source code.

Regarding dependent claim 17, Kuznetsov does not explicitly teach that **the data representation format comprises expanded XML entities, deleted XML declarations and DOM model data converted to Xpath model data**.

ADO.NET teach that Xpath document is created and used to expand the items in the xml data store so that they can be transformed using an XSLT (pp 18 – 19), which meets the limitation of **the data representation format comprises expanded XML entities, deleted XML declarations and DOM model data converted to Xpath model data**.

It would have been obvious to one of ordinary skill in the art at the time of the invention to combine the teachings of Kuznetsov with those of ADO.NET because such a combination would provide the users of Kuznetsov with the benefit of explicit implementation of XPath via source code.

Regarding dependent claim 18, Kuznetsov does not explicitly teach that **the optimized data store is an XpathDocument**.

ADO.NET teach that Xpath document is created and used to store and manipulate the xml data set (pp 18 – 19), which meets the limitation of **the optimized data store is an XpathDocument**.

It would have been obvious to one of ordinary skill in the art at the time of the invention to combine the teachings of Kuznetsov with those of ADO.NET because such a combination would provide the users of Kuznetsov with the benefit of explicit implementation of XPath via source code.

Regarding claim 38, the claim incorporates substantially similar subject matter as claim 12 and is rejected along the same rationale.

(10) Response to Argument

Appellant argues that Kuznetsov fails to teach **the subset of the one or more XML items is less than the whole one or more input XML items** because Kuznetsov does not explain exactly how to translate a subset (p 5 and 6, Bullet a).

The Office disagrees.

While the Office maintains that Kuznetsov does not exactly say that **the subset of the one or more XML items is less than the whole one or more input XML items**, Kuznetsov does teach that any number of translators can be implemented simultaneously, such that an entire set (***or selected subset***) of packets can be translated during runtime (Column 13, line 66 – column 14, line 1). Thus, it is the contention of the Office that the skilled artisan would have been able to deduce that 'one' is a number and that the selected subset is less than the entire set because the use of alternative language, i.e. 'or', implies that the entire set is different from a selected subset. Therefore, one possible implementation taught by Kuznetsov would be the use of 'one' (any number) translator to translate a 'selected subset' (as oppose to the entire set).

Appellant argues that Kuznetsov is not an enabling reference because certain subject matter such as 'subset' is not enabled (pp 7 and 8, bullet b).

The Office disagrees.

Firstly, appellant provides no evidence or citations for the assertion(s) made against the reference. Thus, the examiner is unable to rebut Appellant's argument.

Secondly, MPEP 2121 states:

> I. PRIOR ART IS PRESUMED TO BE OPERABLE/ENABLING
> When the reference relied on expressly anticipates or makes obvious all of the elements of the claimed invention, the reference is presumed to be operable. Once such a reference is found, the burden is on appellant to provide facts rebutting the presumption of operability. In re Sasse, 629 F.2d 675, 207 USPQ 107 (CCPA 1980).
>
> See also MPEP § 716.07, which states:
> Since every patent is presumed valid (35 U.S.C. 282), and since that presumption includes the presumption of operability (Metropolitan Eng. Co. v. Coe, 78 F.2d 199, 25 USPQ 216 (D.C.Cir. 1935), examiners should not express any opinion on the operability of a patent. Affidavits or declarations attacking the operability of a patent cited as a reference must rebut the presumption of operability by a preponderance of the evidence. In re Sasse, 629 F.2d 675, 207 USPQ 107 (CCPA 1980).
>
> III. EFFICACY IS NOT A REQUIREMENT FOR PRIOR ART ENABLEMENT
> A prior art reference provides an enabling disclosure and thus anticipates a claimed invention if the reference describes the claimed invention in sufficient detail to enable a person of ordinary skill in the art to carry out the claimed invention; "proof of efficacy is not required for a prior art reference to be enabling for purposes of anticipation." Impax Labs. Inc. v. Aventis Pharm.Inc., 468 F.3d 1366, 1383, 81 USPQ2d 1001, 1013 (Fed. Cir. 2006). See also MPEP § 2122

The examiner notes that appellant has not rebutted the presumption of validity for the cited reference.

Lastly, MPEP 2121.01 states:

> II. 35 U.S.C. 103 REJECTIONS AND USE OF INOPERATIVE PRIOR ART
> "Even if a reference discloses an inoperative device, it is prior art for all that it teaches." Beckman Instruments v. LKB Produkter AB, 892 F.2d 1547, 1551, 13 USPQ2d 1301, 1304 (Fed. Cir. 1989). Therefore, "a non-enabling reference may qualify as prior art for the purpose of determining obviousness under 35 U.S.C. 103." Symbol Techs. Inc. v. Opticon Inc., 935 F.2d 1569, 1578, 19 USPQ2d 1241, 1247 (Fed. Cir. 1991).

Thus, the examiner need not rebut Appellant's argument.

Appellant argues that the proposed modification of Kuznetsov to meet the claim would render appellant's claimed invention inoperable (pp 8 – 10, bullet c).

The Office disagrees.

First, it should be noted that the Office does not fully understand the argument. By appellant's own quotation, which unfortunately has no citation, the MPEP states:

> If a reference is cited that requires some modification in order to meet the claimed invention or requires some modification in order to be properly combined with another reference and such modification destroys the purpose or function of *the invention disclosed in the reference*, one of ordinary skill in the art would not have found a reason to make the claimed modification. In re Gordon, 733 F.2d 900, 221 USPQ 1125 (Fed. Cir. 1984).

Simply stated, the proposed modification of Kuznetsov does NOT render the invention disclosed by Kuznetsov inoperable; thus, the argument is moot.

Appellant argues that Omoigui fails to teach that **the transformer comprises an action frame stack that holds one or more actions, an event state machine that tracks state associated with transforming the one or more input XML items and an event processor that receives events generated in processing the one or more actions stored in the action frame stack** because the action stack can be considered

analogous but not equivalent to a queue that can store actions (pp 10 – 13, Bolded Bullet B).

The Office disagrees.

First, by appellant's own admission, an action stack can be considered analogous, although not equivalent, to a queue that can store actions while other actions are being processed (p 13, second paragraph). It is respectfully submitted that the 'queue' disclosed by Omoigui is analogous to the claimed action frame stack and thus meets the limitation within the broadest, reasonable interpretation in light of the specification as required under 35 USC 103.

Appellant argues that the ADO.NET reference should not be afforded its rightful date because the date the reference was published or placed on the Internet is unknown (p 13 – 15, bullet a).

The Office disagrees.

As explained to appellant previously, the reference consists of slides that were presented at a conference that was held March 6 – 8, 2001 in Copenhagen and presumed by appellant (p 14, second paragraph), which antedates appellant's effective filing date. MPEP 2128.01 states:

> IV. PUBLICLY DISPLAYED DOCUMENTS CAN CONSTITUTE A "PRINTED PUBLICATION" EVEN IF THE DURATION OF DISPLAY IS FOR ONLY **A FEW DAYS** AND THE DOCUMENTS ARE NOT DIS-SEMINATED BY COPIES OR INDEXED IN A LIBRARY OR DATABASE
> A publicly displayed document where persons of ordinary skill in the art could see it and are not precluded from copying it can constitute a "printed publication," even if it is not disseminated by the distribution of reproductions or copies and/or indexed in a library or database. As stated in In re Klopfenstein, 380 F.3d 1345, 1348, 72 USPQ2d 1117, 1119 (Fed. Cir. 2004), "the key inquiry is whether or not a reference has been made 'publicly accessible.'" Prior to the critical date, a fourteen-slide presentation disclosing the

> invention was printed and pasted onto poster boards. The printed slide presentation was displayed with no confidentiality restrictions for approximately three cumulative days at two different industry events. 380 F.3d at 1347, 72 USPQ2d at 1118. The court noted that "an entirely oral presentation that includes neither slides nor copies of the presentation is without question not a 'printed publication' for the purposes of 35 U.S.C. § 102(b). Furthermore, a presentation that includes a transient display of slides is likewise not necessarily a 'printed publication.'" 380 F.3d at 1349 n.4, 72 USPQ2d at 1122 n.4. In resolving whether or not a temporarily displayed reference that was neither distributed nor indexed was nonetheless made sufficiently publicly accessible to count as a "printed publication" under 35 U.S.C. 102(b), the court considered the following factors: "the length of time the display was exhibited, the expertise of the target audience, the existence (or lack thereof) of reasonable expectations that the material displayed would not be copied, and the simplicity or ease with which the material displayed could have been copied." 380 F.3d at 1350, 72 USPQ2d at 1120. Upon reviewing the above factors, the court concluded that the display "was sufficiently publicly accessible to count as a 'printed publication.'" 380 F.3d at 1352, 72 USPQ2d at 1121.

Appellant generally argues claims 6, 8, 9 and 10, claim 12, and claims 15, 16, and 18 as not being taught because the reference lacks certain key words and/or ADO.Net does not provide enough details of the claimed functionality (pp 16 – 18).

Conveniently, appellant fails to mention that claims 11, 14 and 18 respectively provide evidence that claims 6 – 10, claims 12 and 13, and claims 15 – 17 are met by the reference because ADO.NET teaches an XpathNavigator, an XPathNodeIterator, and an XpathDocument, respectively, which is all the claims require.

Specifically, dependent claims 6, 8, 9 and 10 seek to further define an input abstractor and dependent claim 11 recites that the input abstractor is an XPathNavigator, which ADO.NET discloses (p 19). While appellant may disclose ‘sample’ code as a possible example in the specification, there is no purposeful and deliberate definition stated in the Specification as to some special XPathNavigator. Consequently, XPathNavigator is essentially a term of art and known element with which the skilled artisan is already familiar. Simply, the skilled artisan is well aware of the functionality of an XPathNavigator and could easily make or, worst case, modify it to

meet the claims under 35 USC 103(a). However, original claim 11, which is part of Appellant's disclosure, provides more than adequate evidence that any XPathNavigator has the functionality expressed in its parent claims 6 – 10.

Similarly, dependent claim 12 seeks only to further define a node selection abstractor and dependent claim 14 recites that the node selection abstractor is an XPathNodeIterator, which ADO.NET discloses (pp 18 and 19). Thus, original claim 14, which is part of Appellant's disclosure, provides more than adequate evidence that any XPathNodeIterator has the functionality expressed in its parent claims 12 and 13.

Similarly, dependent claims 15 and 16 seeks only to further define an optimized data store and dependent claim 18 recites that the optimized data store is an XPathDocument, which ADO.NET discloses (pp 18 and 19). Thus, original claim 18, which is part of Appellant's disclosure, provides more than adequate evidence that any XPathDocument has the functionality expressed in its parent claims 15 – 17.

Lastly, for completeness in regards to claim 38, the argument(s) and thus rebuttal for claim 38 has been addressed in connection with claim 12. In addition, appellant argues that it is unclear how the subset is dynamically constructed (p 17, last tow paragraphs).

Simply, appellant provides no support for what is meant by dynamically constructed. The skilled artisan is well aware that a query is used to gather certain data; one is usually unaware what this data is, which is why one queries it. However,

whatever the data happens to be as a result of the query this data is used, which is the very definition of dynamic construction.

(11) Related Proceeding(s) Appendix

No decision rendered by a court or the Board is identified by the examiner in the Related Appeals and Interferences section of this examiner's answer.

For the above reasons, it is believed that the rejections should be sustained.

Respectfully submitted,

/Nathan Hillery/

Examiner, Art Unit 2176

Conferees:

/DOUG HUTTON/
Supervisory Patent Examiner, Art Unit 2176

/Stephen S. Hong/
Supervisory Patent Examiner, Art Unit 2178